LIDF: Layered Intrusion Detection Framework for Ad-Hoc Networks

Nikos Komninos[1] and Christos Douligeris[2]

[1]*Algorithms and Security Group, Athens Information Technology, GR-190 02 Peania Greece,* nkom@ait.edu.gr
[2]*Department of Informatics, University of Piraeus, GR-185 34 Piraeus Greece*
cdoulig@unipi.gr

Abstract—As ad-hoc networks have different characteristics from a wired network, the intrusion detection techniques used for wired networks are no longer sufficient and effective when adapted directly to a wireless ad-hoc network. In this article, first the security challenges in intrusion detection for ad-hoc networks are identified and the related work for anomaly detection is discussed. We then propose a layered intrusion detection framework, which consists of collection, detection and alert modules that are handled by local agents. The collection, detection and alert modules are uniquely enabled with the main operations of ad-hoc networking, which are found at the OSI link and network layers. The proposed modules are based on interpolating polynomials and linear threshold schemes. An experimental evaluation of these modules shows their efficiency for several attack scenarios, such as route logic compromise, traffic patterns distortion and denial of service attacks.

Index Terms— ad-hoc networks, collection, detection and alert modules, framework.

1. Introduction

An ad-hoc network is a collection of nodes that can be rapidly deployed as a multi-hop packet radio network without the aid of any existing network infrastructure or centralized administration. Therefore, the interconnections between nodes are capable of changing on a continuous and arbitrary basis. Unlike networks using dedicated nodes to support basic functions like packet forwarding, routing, and network management, in ad-hoc networks these functions are carried out by all available nodes [11, 12]. Applications of ad-hoc networks range from military tactical operations to civil rapid development, such as emergency search-and-rescue missions, data collection/sensor networks, and instantaneous classroom/meeting room applications.

The nature of the ad-hoc environment makes it vulnerable to an adversary's malicious attacks. Such networks are susceptible to attacks ranging from passive eavesdropping to active interfering. Unlike wired networks where an adversary must gain physical access to the network wires or pass though several lines of defense at firewalls and gateways, attacks on a wireless network can come from any direction and target all nodes. Therefore ad-hoc networks do not have a clear line of defense, and

every node must be prepared for encounters with an adversary directly or indirectly [11, 12].

In ad-hoc networks nodes are receptive to being captured, compromised, and hijacked since they are units capable of roaming independently. Since tracking down mobile nodes is difficult to achieve, attacks by compromised nodes are far more damaging and much harder to detect. Therefore, the nodes and the network infrastructure must be prepared to operate in a non-trusting mode. Furthermore, the lack of a centralized authority gives ground to adversaries to exploit new types of attacks and to break the required for efficient operations cooperative algorithms.

In this article, we propose a layered intrusion detection framework (LIDF) to detect compromised and malicious nodes in an ad-hoc network. LIDF is enabled with the main operations of ad-hoc networking, which are found at the Open System Interconnection (OSI) link layer with *one-hop connectivity* / *frame transmission* and network layer with *routing* / *data packet forwarding*. LIDF consists of collection, detection and alert modules that operate locally in every node in an ad-hoc network. These modules collect, detect and inform neighboring nodes for their possible compromised status. The collection and storage of audit data is performed with the use of a binary tree. The detection is achieved with Lagrange interpolating polynomials and the alert is accomplished with linear threshold schemes. Experimental results prove the effectiveness of our approach.

Following this introduction, this article is organized as follows. Section 2 presents an introduction to intrusion detection and focuses on anomaly detection as the most related work that applies to ad-hoc networks. Section 3 describes the detection framework and discusses how the collection, detection and alert modules operate. Section 4 presents and discusses the experimental results. Finally, section 5 concludes the article with a review of our contribution and suggestions for future research.

2. Intrusion Detection Challenges

When a set of actions that attempt to compromise the integrity, confidentiality, or availability of a mobile node takes place, intrusion prevention techniques, such as encryption and authentication, are usually the first line of defense. However, intrusion prevention alone is not sufficient when systems become more complex and as security is often the after-thought. There are always weaknesses in the systems due to design and programming errors, or various “socially engineered” penetration techniques [2, 3, 12].

For example, even though exploitable “buffer overflow” security holes, which can lead to an unauthorized root shell, were first reported many years ago they still exist in some recently released system software. Furthermore, as illustrated by the Distributed Denial-of-Services (DDoS) attacks launched against major Internet sites where security measures were in place, the protocols and systems that are designed to provide services are inherently subject to these attacks [6, 9]. Intrusion detection can be used as a second wall to protect network systems because once an intrusion is detected, a response can then be put into place to minimize damages.

By definition, intrusion detection involves capturing data and reasoning about the evidence in the data to determine whether the system is under attack [12, 15]. The most important difference between fixed and ad-hoc networks is perhaps that the latter do not have a fixed infrastructure. Compared to wired networks where traffic monitoring is usually done at switches, routers and gateways in a network-based intrusion detection system (IDS), the mobile ad-hoc environment does not have such traffic concentration points and therefore only host-based IDS can be used.

While network-based IDS listen on the network, capture and examine individual packets flowing through a network, host-based IDS [8, 12, 15, 25] are concerned with what is happening on each individual node. They are able to detect actions such as repeated failed access attempts or changes to critical system files, and they normally operate by accessing log files or monitoring real-time system usage.

Intrusion detection techniques are categorized into misuse detection and anomaly detection [17, 18]. Misuse detection bases its idea on precedence, rules and misuse detectors that look for behavior which matches the already known attack scenario. A typical misuse detection system takes in audit data for analysis and compares these data to large databases of attack signatures. The attack signatures, or known attack patterns, are normally specified as rules with respect to timing information. If any comparison between the audit data and the known attack patterns results in a match, an intrusion alarm setting sounds. This type of detection systems is as good as the database of attack signatures that it uses to compare to.

Furthermore, misuse detection systems use patterns of well-known attacks or weak spots of the system to match and identify known intrusions [21]. For example, a signature rule for the “guessing password attack” can be “there are more than 4 failed login attempts within 2 minutes”. The main advantage of misuse detection is that it can accurately and efficiently detect instances of known attacks whereas its main disadvantage is that it lacks the ability to detect the newly invented attacks.

Anomaly detection bases its idea on statistical behavior. Anomaly detectors look for behavior that deviates from normal system use [26, 27]. A typical anomaly detection system takes in audit data for analysis. The audit data is transformed to a format statistically comparable to the profile of a user. Initially, the user's profile is generated dynamically by the system and it is subsequently updated based on the user's usage. Thresholds are always associated to all the profiles. If any comparison between the audit data and the user's profile results in a deviation that crosses a set threshold, an intrusion alarm is set. This type of detection systems is well suited to detect unknown or previously not encountered attacks [15, 18, 20].

For example, the normal profile of a user may contain the averaged frequencies of some system command used in his or her login sessions. If for a session that is being monitored, the frequencies are significantly lower or higher, then an anomaly alarm will be raised. The main advantage of anomaly detection is that it does not require prior knowledge of intrusion and it can, thus, detect new intrusions whereas its main disadvantage is that it may not be able to describe what the attack is and it may have a high false positive rate.

2.1. Related Work

The intrusion detection techniques that are presented in this section are chosen due to their suitability for anomaly detection. Anomaly detection is the main approach for intrusion detection in ad-hoc networks because in this new environment intrusions will come in the form of new attack, i.e. attacks that have not yet been defined. Moreover, specification-based anomaly detection [28] is a hybrid combination of anomaly-detection and knowledge-based intrusion detection techniques that mitigate the weaknesses of the two approaches while magnifying their strengths. This method uses a logic-based description of expected behavior to construct a profile based on human behavior or expertise. The authors in [29] built application software for online attack identification without debilitating waits for anti-virus updates or software patches.

In ad-hoc networks, intrusion detection is based on statistical anomaly detection, rather than misuse detection, because of the perceived difficulties of continually updating misuse detection rules (or signatures). If an intrusion warrants a broader investigation, nodes are expected to trigger the cooperation of other nodes for global-scale intrusion detection. A likely algorithm for performing this task collects observed data from all the nodes about the suspected node, and then weighs the majority consensus to determine whether an intrusion has occurred. In [27], for example, each

node concurrently runs a software agent that monitors its own system activities as well as traffic among neighboring nodes within its radio range. Each node also analyzes its own data for local intrusion detection.

The authors of [13] proposed the idea of ad-hoc nodes monitoring their neighboring nodes' packet-forwarding behavior in what they called a watchdog process. After a node forwards a packet, the watchdog monitors the next node to verify that the packet is forwarded again. This scheme assumes source routing with each packet carrying its route information so that the watchdog knows a tracked packet's proper route. If a watchdog observes that a neighboring node drops more packets than a given threshold, the node is deemed to be misbehaving.

In the systems presented in [1] and [19] each node maintains a "malcount" for neighboring nodes, i.e. the number of observed occurrences of misbehavior. When a node's malcount exceeds a given threshold, its neighbors send out an alert to the other nodes, which then check their malcounts for the suspected node. If a suspected node triggers two or more alerts, it is deemed to be malicious. Naturally, this scheme works only if at least two trustworthy nodes observe a suspected node; it can fail if malicious nodes send out false alerts.

In [4] the Confidant scheme was proposed, which, similar to the previous approaches, relies on ad-hoc nodes to monitor their neighboring nodes' routing behavior. Source routing is assumed, so that nodes know the correct route for tracked packets. Confidant's innovation is a reputation system that works with network monitoring. This system consists of a table of observed nodes and their reputation ratings. If a node is observed to be misbehaving (deviating from its expected routing behavior), the reputation system changes the node's rating by a weighting function depending on the new observation's trustworthiness.

The Mobile Intrusion Detection System (MobIDS) [8, 10, 17] is similar to the other schemes described here. Multiple sensors in the ad-hoc network keep track of observed instances of the nodes' behavior. In MobIDS, though counts from multiple sensors are combined with a weighting function reflecting the different sensors' credibility to create a local rating for a suspect node. These local ratings are then distributed periodically via broadcasting to the neighboring nodes. Each node averages the local ratings it receives into global ratings for other nodes. Nodes are deemed to be misbehaving if their ratings drop below a given threshold.

There are certain difficulties in realizing all these schemes. First, there is no clear separation between normalcy and anomaly in a mobile environment. A node that sends out false routing information could be the one that has been compromised, or merely the one that is temporarily out of synchronization due to a likely volatile physical movement Second, these schemes are only useful to prevent intruders from the outside (external attacks) and they are not useful when an internal node is compromised (internal attack). Third, due to the bandwidth limitations, battery constraints and frequent disconnects, users often adopt new operation modes such as disconnected operations. This suggests that existing anomaly detection models may not be able to determine that such new operations are certified and subsequently identify them as intrusions.

3. Layered Intrusion Detection Framework

The layered intrusion detection framework (LIDF) presented in this article is designed especially for ad-hoc networks by taking into considerations the characteristics of ad-hoc networks and the problems that existing IDS systems face when deployed in a wireless environment. The dynamic and cooperative nature of ad-hoc networks suggests that LIDF should be designed in a dynamically and cooperative fashion. In a wireless environment each node should have its own intrusion detection engine to help it perform intrusion detection since it cannot rely on other nodes that may leave the network at anytime. Ad-hoc networks also do not have traffic concentration points that allow for intrusion detection at a centralized location, a fact that further emphasizes the need for each node to have it own intrusion detection module.

Similar to [8, 15, 25], intrusion detection is to be performed locally via a local agent on each node utilizing the partial, localized audit data since this is the most reliable source of audit data for a node. Each node can then perform cooperative intrusion detection when more information is required from other nodes to confirm the intrusion. For cooperative intrusion detection, the individual node is required to work with neighboring nodes to gather more audit data for intrusion detection. This suggests that there should be a secure communication channel between the nodes participating in the cooperative intrusion detection.

LIDF should be interoperable with existing intrusion detection systems, since an ad-hoc network can be deployed in an environment that contains different types of networks (e.g. a university campus), which are interconnected and have already existing

intrusion detection systems running on them. Allowing the exchange of audit data and other information between the different systems may increase the overall effectiveness of intrusion detection in the entire environment.

As illustrated in Figure 1, LIDF consists of the following components; a collection module, a detection module and an alert module. The collection module collects data at the OSI link and network layers. Information in needed from both these two different layers to perform layered intrusion detection. Layered intrusion detection is necessary as certain attacks that target the upper layer may seem perfectly legitimate to the lower layers. For example, a DDoS attack that targets the network layer of an ad-hoc network seems legitimate to the link layer that handles node connectivity.

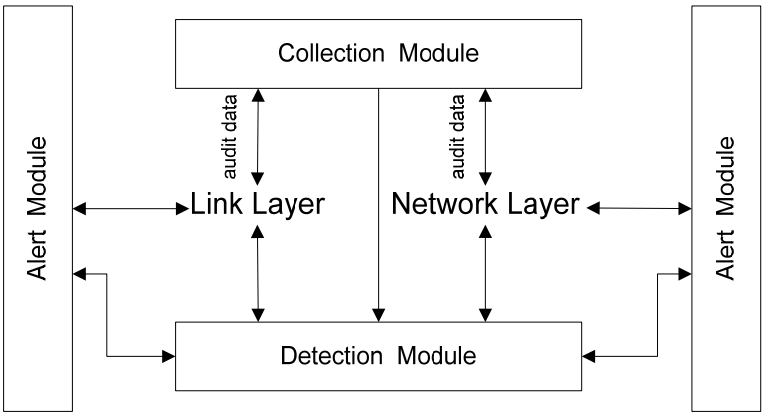

Fig. 1 – Intrusion Detection Framework

In mobile nodes intrusion detection should be done on the basis of different levels of escalation, starting from the simplest and least battery consuming intrusion detection operation to the more complex and CPU intensive operations. The detection module processes the most relevant audit data collected from the different layers based on the mode that the mobile node is currently operating in.

In an anomaly detection scenario, when an intrusion is detected, the system needs to respond immediately locally on the neighboring hosts. The neighboring nodes can then respond to the intrusion either locally or cooperatively. The alert module is also necessary when the node needs to perform intrusion detection cooperatively as well as when it has to sound a global alarm to the ad-hoc network. This can be achieved with the main protocol operations of ad-hoc networking.

Our proposed intrusion detection framework is closely related to the main operations of ad-hoc networking, which mainly take place at the OSI link layer (*one-hop connectivity* and *frame transmission*) and at the OSI network layer (*routing* and

data packet forwarding) [11]. Data link layer protocols maintain connectivity between neighboring nodes and ensure the correctness of frames transferred, whereas routing protocols exchange routing data between nodes and maintain routing states at each node accordingly. Based on the routing states, data packets are forwarded by intermediate nodes along an established route to the destination.

These operations comprise of link and network operations that integrate a framework for local and cooperative intrusion detection. When link and/or network operations take place in the ad-hoc network, the data collection, detection and alert modules presented in detail in the next sections are enabled. In particular, when a new neighbour is detected by a link layer protocol and when routing states are updated by a network layer protocol of a node, the collection, detection and alert modules are enabled.

Summarising, once the data is collected (section 3.1) a unique polynomial is generated with Lagrange interpolation techniques. Next, a secret function is selected based on the unique polynomial and the detection module checks whether the secret function converges at point of intersection with the specified polynomial. If it converges the node is considered to be compromised, else valid (section 3.2). In addition, the response module broadcasts to the neighbouring nodes the secret function, the shares and the polynomial coefficients using discrete logs and linear threshold schemes so as nodes can check the status of the source node.

3.1. Collection Module

The data collection module aims at collecting audit data in a coordinate format, (x, y), to create a tree structure and at exhibiting the existence of a polynomial for interpolation with unevenly spaced data. Data are aggregated into pairs that describe the changes in the routing table of a node. Thus, we have defined a two-dimensioned coordinate system (x_i, y_i), where the x_i – coordinate denotes the i^{th} data value, or point, of route cashes (i.e. specific routes to neighbouring nodes) and the y_i – coordinate denotes the i^{th} data point of traffic patterns (i.e. number of packets forwarded).

As mentioned in the previous section, the data are locally collected in every node when *one-hop connectivity* and/or *routing* are taking place in each node. Thus, data may contain values that are not equivalently spaced as the result of observations. Assuming that the x-coordinates of the points are distinct a unique interpolating polynomial between the points will always exist.

The assertion of our assumption for distinct points can be guaranteed with a binary tree structure, which collects and stores the data as illustrated in Figure 2. Figure 2 shows an example of a tree structure with 11 elements. Since the structure is a tree, every node is either a leaf or a parent of one or two children nodes. The root is at level $l = 0$ and its height is h, where $h = 4$.

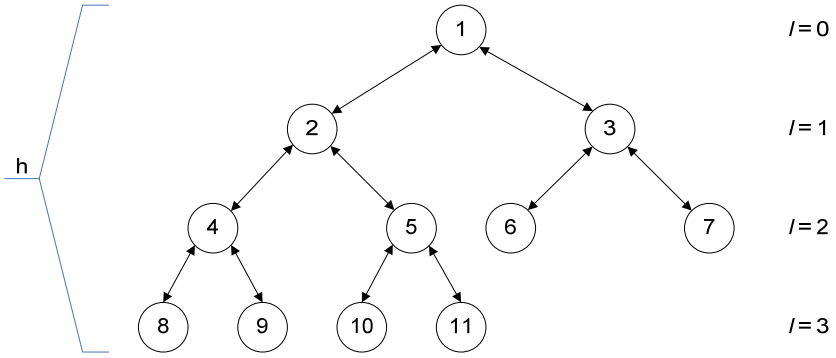

Fig. 2 – An example of the binary tree structure

When a node is connected to an ad-hoc network, it creates a local tree with only 3 members; the root and its two children. The left leaf (number 2) of the root is activated whereas the right leaf (number 3) is deactivated, when frame transmission is taking place. Then, the next audit pair of data (x_i, y_i) and (x_j, y_j) for $x_i \neq x_j$ is stored in the new leaves at positions 4 and 5 and so forth. Furthermore, the right leaf of the root with number 3 is activated, whereas the left leaf is deactivated when routing and packet forwarding take place in the ad-hoc network. Similarly, the pair of data (x_i, y_i) and (x_j, y_j) for $x_i \neq x_j$ are stored in the leaves with numbers 6 and 7. For $x_i = x_j$, the x_i coordinate is increased by one, i.e., $x_i = x_j + 1$ and stored in the tree.

For example, the audit pair data (x_i, y_i) retrieve their values from the percentages of routing and traffic parameters (i.e. change of node distance (DIST); change of route entries (PCR); change of traffic (PSTC); change of number of hops (PCH); change of bad routes (PCB); change of updated routes (PCU); and change of stale routes(PCS)). Each time left or right leaf is activated (x_i, y_i) and (x_j, y_j) are assigned values based on the equations 1 and 2.

$$x_i = \text{PCR} + \text{PSTC} + \text{PCH} \ mod\ 101 \quad (1)$$

$$y_i = \text{PCB} + \text{PCU} + \text{PCS} \ mod\ 101 \quad (2)$$

Based on equations 1 and 2, we have collected the following unique data points at tree level $l = 2$ of figure 2: (21, 54), (43, 45), (73, 48), and (91, 27). When data from both layers are collected, the detection module is activated and it then processes the data from the tree.

3.2. Detection Module

The collected data of the collection module form a set of data points which represent a unique polynomial. The data points that have created the binary tree of Figure 2 are uniquely interpolated between those data points. The detection module selects the most recent pair data that have been collected when the link and network operations were enabled. In figure 2, for example, the detector will use only the data from the second level ($l = 2$) with $h = 3$ to construct a unique polynomial. Data in the third level ($l = 3$) with $h = 4$ do not form a representative data sample since they had been collected only when link layer operations were taken place.

The detection module checks whether the constructed polynomial converges in a specified interval (i.e. defined by a secret function). If it converges, the node is considered to be compromised else the node is assumed valid. A mathematical description and notation follows in the next paragraphs.

By definition, a polynomial interpolation is the interpolation of a given data set by a polynomial [14]. Given a set of $n + 1$ data points (x_i, y_i) where no two x_i are the same, we are looking for a polynomial p of degree at most n with the property $p(x_i) = y_i,\ i = 0,...,n$. Suppose that the interpolation polynomial we create from the tree is in the form:

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_2 x^2 + a_1 x + a_0 \tag{3}$$

The statement that p interpolates the data points means that $p(x_i) = y_i,\ \forall i \in \{0,1,...,n\}$, which in matrix-vector is given by:

$$\begin{bmatrix} x_0^n & x_0^{n-1} & x_0^{n-2} & \dots & x_0 & 1 \\ x_1^n & x_1^{n-1} & x_1^{n-2} & \dots & x_1 & 1 \\ \vdots & \vdots & \vdots & & \vdots & \vdots \\ x_n^n & x_n^{n-1} & x_n^{n-2} & \dots & x_n & 1 \end{bmatrix} \begin{bmatrix} a_n \\ a_{n-1} \\ \vdots \\ a_0 \end{bmatrix} = \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_n \end{bmatrix} \tag{4}$$

which expresses a system of linear equations in the coefficients a_n. We need to solve this system for a_n to construct the interpolant $p(x)$. Since the matrix on the left which is

commonly referred to as a Vandermonde matrix has non-zero determinant [14], it can be easily shown that there exists a unique interpolating polynomial.

In order to find the coefficients a_n for the interpolating polynomial we must solve the above matrix equation in the vector space $\prod_n$, which is the vector space of polynomials of degree n. Since this is a costly operation in clock cycles and considering that the detection module is encapsulated in wireless nodes of an ad-hoc network we construct an interpolation polynomial in the Lagrange form [14]. (Note that all mathematical operations are performed in the finite field of integers $\mathbb{Z}_q$ (where q is a prime)).

The interpolation polynomial $L(x)$ can be mathematically represented as a linear combination:

$$L(x) = \sum_{j=0}^{n} y_j l_j(x) \mod q \tag{5}$$

of Lagrange basis polynomials:

$$l_j(x) = \prod_{i=0, j\neq i}^{n} \frac{x - x_i}{x_j - x_i} \mod q \tag{6}$$

given a set of n+1 data points (x_i, y_j) where the x_i values must be distinct. The interpolation polynomial in Lagrange form for a specific number of points n = 2, 3, 4, 5 in the form (x_i, y_j) can be graphically illustrated as in Figure 3.

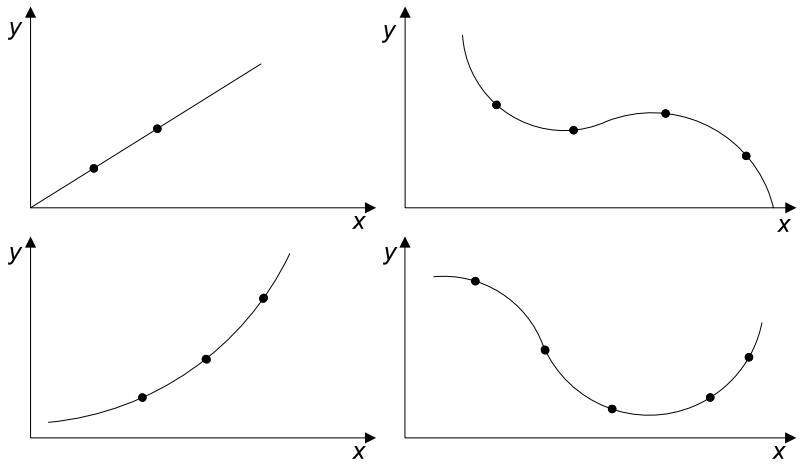

Fig. 3 – Interpolation polynomials in Lagrange form

In the detection module, it is essential to discover for which $(x_1, \ldots, x_i)$ the sequence of interpolating polynomials uniformly converges. Particularly, it is necessary to determine for which pairs (x_i, y_j) L (x) converges. If the convergent points are found, the detection

module can identify the status (i.e. compromised) of the node. A node is considered to be compromised if the convergence points (x_i, y_j) of L (x) fall under the right side of a secret function f (x), which is defined by the detection module (see Figure 4).

It is known that for any function f (x) continuous on an interval [a, b] $\in Z_q$ there exists a table of x_i for which the sequence of interpolating polynomials $L(x)$ converges to f (x) uniformly on [a, b]. This is due to the Weierstrass approximation and the Chebyshev alternance theorems [14]. The Weiestrass approximation theorem states that the sequence of polynomials of best approximation $p_n^*(x)$ converges to f (x) and the Chebyshev alternance theorem states that such polynomials intersect f (x) at least n + 1 times. Choosing the points of intersection as interpolation points, we obtain the interpolating polynomial $L(x)$ in its Lagrange form. This polynomial coincides with the best approximation polynomial.

Let us define a linear secret function $f(k)$ to be in set K of secrets $k \in K$ that intersects $L(x)$ in at least n points:

$$f(k) = a^{n-1}k + \sum_{i=1}^{n-2} a_i \tag{7}$$

where $a_1, a_2, ..., a_{n-1} \in Z_p$ are the coefficients of the polynomial $L(x)$ of degree n-1. Since $L(x)$ is of degree n-1, it intersects $f(k)$ in n points.

Next, it is essential to determine when $L(x)$ converges. Based on the convergence theorem [14], $L(x)$ converges uniformly in [a, b] if there exists $a < \varepsilon < b$ such that $|L(x) - f(k)| < \varepsilon$. An example of $L(x)$ and $f(k)$ is depicted in Figure 4.

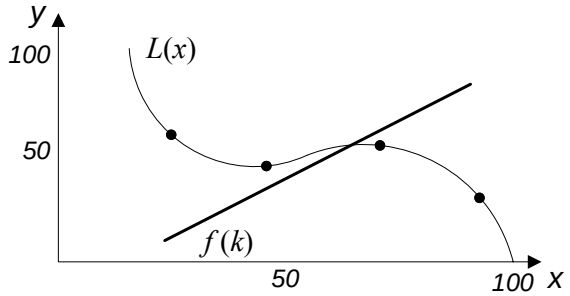

Fig. 4 – Uniformly convergence of *L*(*x*) with respect to *f* (*k*)

In Figure 4, the polynomial $L(x)$ is graphically represented by the curve whereas the secret function f (k) is presented by the straight line. In the example of section 3.1 we have collected the unique data points (21, 54), (43, 45), (73, 48), (91, 27) and thus, we can easily substitute and calculate equations 5 and 6 for q = 101. While our detection

module "observed" the normal behaviour of the network, including node joining, leaving and speeding in the ad hoc network, function $f(k)$ was calculated by equation 7:

$$f(k) = 4/3k + 20 \tag{8}$$

Once the detection module has determined whether the node has been compromised or not, the alert module is responsible for distributing the node's status to the neighbouring nodes.

3.3. Alert Module

When each node is connected for the first time to an ad-hoc network, the data collector passes the audit data to the detector and the alert module informs its neighboring nodes that the new entering node has been compromised. At this stage, every new entering node is considered to be compromised unless proven otherwise. A node's compromised status does not allow gaining access to specific applications or services in the ad-hoc network.

The proposed alert module enables nodes to perform cooperatively intrusion detection as well as to carry out a global alarm to the ad-hoc network with the use of a linear threshold scheme. Linear threshold schemes have been mainly applied in the distribution of shares of a secret to a set of shareholders such that the secret is a linear combination of the shares [14]. The necessary mathematical notation follows in the next paragraphs.

An (m, n) linear threshold scheme distributes a secret to a set of n shareholders, or nodes in the ad-hoc context, such that the secret is a linear combination of the shares of any m nodes. In equation 7, we have defined the secret function $f(k)$, which determines whether a node has been compromised or not, to be in a set K of secrets, $k \in K$, and we have assumed that each node i is in the set P ($|P| = n$) of nodes. Similar to [14] in order to distribute $f(k)$, we generate a share s_i, which is constructed with an index shift by equations 5 and 7, for $i \in P$ with a polynomial $a(i)$:

$$s_i = f(k) + \sum_{l=1}^{m-1} a_l i^l \tag{9}$$

where s_i is in the set S_i of shares, and S_i is in the set S of share sets. For linear threshold schemes, $S_i = S_j$ for all $i, j \in P$ [14]. To reconstruct $f(k)$, we combine s_i for all i in an authorized subset B ($|B| = n$) of P and use Lagrange interpolation:

$$f(k) = \sum_{i \in B} \psi_i(s_i) \qquad \text{where} \qquad \psi_i = \prod_{l \in B, l \neq i} \frac{l}{l-i}. \tag{10}$$

ψ_i is a homomorphism from S_i to K. For linear threshold schemes, the homeomorphisms are multiplications by a scalar ψ_i. In addition, we utilize a homomorphic commitment function $C(x)$ that maps from the plaintext to the ciphertext and is hard to invert. Assuming that the computation of discrete logs in a finite field is intractable, we use the commitment function:

$$C(x) = g^x \tag{11}$$

where g is a generator of Z_p:

$$\forall b \in \{1,..., p-1\} \exists a \in \{1,..., p-1\} : g^a \equiv b \bmod p \tag{12}$$

The alert module broadcasts the commitment to the secret $g^{f(k)}$, the shares g^{s_i}, and the coefficients of the polynomials $g^{a_1}...g^{a_{(m-1)}}$. The nodes $j \in P$ then verify that:

$$g^{s_i} \equiv g^{f(k)} \prod_{i=1}^{m-1} (g^{a_i})^{i^l} \tag{13}$$

for each $i \in B$, and that:

$$g^{f(k)} \equiv \prod_{i=1} (g^{s_i})^{\psi_i} \text{, where } \psi_i \equiv \prod_{l \in B, l \neq i} \frac{l}{l-i}. \tag{14}$$

The alert protocol can also be used for cooperative intrusion detection where the individual node is required to work with neighboring nodes to gather more audit data for detection purposes. In the same manner as before, audit data can be distributed among the nodes based on the alert protocol. However, the corporate intrusion detection requires each node to have at least three neighboring nodes for data collection in order to minimize the possibility of fraud by the compromised and/or malicious node(s). For example, a compromised node will send malicious data to the neighboring node in order to fool the detection mechanism. However, the malicious node does not know the secret function and, thus, it can only guess the data with a 50% probability.

To prove that only valid nodes can create a valid subshare, which in our case is another k and, thus, $f(k)$, let us assume that we know the shares s_i of the nodes $i \in B$ and the coefficients of the polynomial a_i used by i to distribute the shares s_i. We could then interpolate the m-1 degree polynomial that another node could have used to distribute shares s'_j of $f(k)$ to n' new nodes $j \in P'$ directly. Suppose that each $i \in B$ broadcasts the same information. Each $j \in P'$ then verifies that s'_j is a valid share of $f(k)$ with the following equation:

$$g^{s'_j} = g^{f(k)} g^{\left(\sum_{i\in B}\psi_i a'_1\right)j} \ldots g^{\left(\sum_{i\in B}\psi_i a'_{m'-1}\right)j^{m'-1}} \quad (15)$$

Equation 15 follows from equations 13, 14 and the homomorphic properties of exponentiation. Since finding discrete logs is intractable, no j can learn $f(k)$ from the broadcast of $g^{f(k)}$. In the check suggested by equation 15, the m nodes $i \in B$ prove that they distributed valid subshares of valid shares to the n' new nodes $j \in P'$. Hence, only a valid node can create a valid subshare. Carrying on the example of sections 3.1 and 3.2 and selecting $k = 12$, we can calculate $f(k)$ (equation 8), generate a share s_i (equation 9) and choose generator $g = 20$ to distribute and to validate $f(k)$ based on equations 14 and 15.

As the number of nodes increases, corporate intrusion detection creates an overhead to the ad hoc network. Considering that each node has at least three neighbouring nodes in current implementation, we found that the subshares distribution triples the network overhead. In corporate intrusion detection, it is not essential nodes to distribute all subshares, which demonstrate their full status, but parts of them following the verification process of equations 13, 14 and 15. Experimental results showed that we could still achieve the detection rates of section 4 (Tables II, III, IV) while distributing half of the states involved with an increase of 1.5 to the total network overhead. Further decrease of the distributed subshares, resulted to a detection rate of less than 90%.

4 Experimental Results

In this study, we evaluate the effectiveness of our approach by considering attacks on routing protocols and specifically the *route logic compromise*, *traffic pattern distortion* and *denial of service* attacks. In the route logic compromise attack, routing information is manipulated by parsing false route messages or by maliciously changing routing cache information. In the traffic pattern distortion attack, packets are maliciously dropped. In the denial of service attack, specific node(s) are unavailable to route information to neighboring nodes. We use data on the node's physical movement and the corresponding change in its routing table as the basis of the trace data.

The routing table change is measured mainly by the percentage of changed routes (PCR) and the percentage of changes in the sum of hops of all routes (PCH). During the tracing process the trace data are gathered for each node and aggregated into a single data set, which describes the changes in the routing tables for all the nodes. The detection module which is learned from this aggregated data set is capable of operating

on any node in the network. A poor performance of the anomaly detection model with a high false alarm rates indicates that the gathered data (including both training and testing processes) are not sufficient and the modeling algorithms need to be refined.

To study the effectiveness of our approach, we have implemented anomaly detection in the *Network Simulator*, *ns*-2, software using three popular proactive and on-demand routing protocols, mainly the Dynamic Source Routing (DSR) protocol [20], the Ad-Hoc On-Demand Distance Vector Routing (AODV) protocol [1, 19], and the Destination-Sequenced Distance-Vector Routing (DSDV) protocol [24, 25].

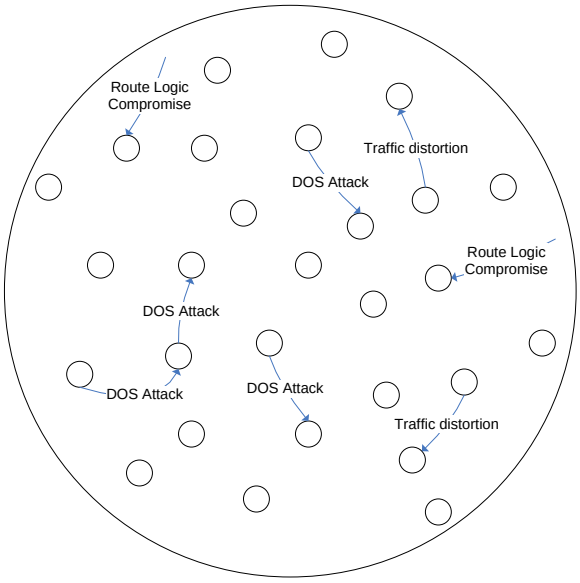

Fig. 5 – Simulation Environment for the Ad Hoc Network

In order to compare among different protocols, we consider the same traffic and topological information but we allow a slight deviation to make maximum utilization of the routing information, since even under the same variables, protocols operate in a slightly different manner. For example, PCH is the percentage of change in the number of total intermediate hops from all source routes cashed in DSR, but the percentage of changes of sum of metrics (i.e. PCB, PCS, and PCU of Table I) to all reachable

TABLE I
PROTOCOL FEATURES

Features	Description
VEL	Velocity
DIST	Change of Distance
PCR	Change of Route entries (%)
PSTC	Change of Traffic (%)
PCH	Change of Number of Hops (%)
DSR, DSDV, AODV	
PCB	Change of Bad Routes (%)
PCU	Change of Updated Routes (%)
PCS	Change of Stale Routes (%)

TABLE II
DETECTION PERFORMANCE ON DSR

Running Time (s)	Detection Rate	False Alarm Rate
100000	95 ± 3.99%	3.451 ± 0.33%
200000	96 ± 3.25%	3.764 ± 0.22%
300000	96 ± 3.56%	2.152 ± 0.75%
400000	97 ± 2.57%	2.326 ± 0.72%
500000	97 ± 2.35%	2.955 ± 0.24%
600000	97 ± 1.32%	2.863 ± 0.74%
700000	98 ± 1.85%	1.486 ± 0.84%
800000	98 ± 1.21%	1.990 ± 0.90%
900000	99 ± 0.45%	0.025 ± 0.12%
1000000	99 ± 0.12%	0.011 ± 0.34%

destinations in DSDV and AODV. Furthermore, our data is collected by route caches, and traffic patterns of each node. The collection modules used a tree structure with a height of h = 20 to store data where the height value was selected based on the performance of the node. The simulation environment is illustrated in figure 5.

Our simulation environment consists of a large number of nodes that are moving in a random walk basis [15]. Some nodes are subject to route logic compromise, traffic pattern distortion and denial of service attacks. In particular, some nodes are considered that their route logic has somehow being compromised. Some other nodes receive distorted traffic and some other nodes are unavailable to routing information. In all three attacks the number of nodes varies that are subject to attacks.

To test our models, we have used several test scripts to generate traces based on their running time. The trace running time varies from 100,000 to 1,000,000 seconds increasing by 100,000 seconds each time (Tables II, III, IV). For each result, we run the simulation 15-times and we report its average and error. In Tables II, III, and IV the experimental results demonstrate that an anomaly detection approach can work well on different ad-hoc networks. Even though the model has been trained with a small trace (100,000 sec), it has already proved satisfactory, so that it is more effective for a much longer trace (1,000,000sec).

TABLE III
DETECTION PERFORMANCE ON AODV

Running Time (s)	Detection Rate	False Alarm Rate
100000	93 ± 4.12%	4.455 ± 0.36%
200000	94 ± 4.43%	4.311 ± 0.67%
300000	94 ± 3.15%	3.689 ± 0.24%
400000	95 ± 3.43%	3.836 ± 0.98%
500000	96 ± 3.78%	3.023 ± 0.56%
600000	97 ± 2.12%	2.287 ± 0.83%
700000	97 ± 2.74%	2.401 ± 0.14%
800000	98 ± 1.45%	1.750 ± 0.98%
900000	98 ± 1.86%	1.425 ± 0.60%
1000000	99 ± 0.01%	0.561 ± 0.12%

TABLE IV
DETECTION PERFORMANCE ON DSDV

Running Time (s)	Detection Rate	False Alarm Rate
100000	93 ± 4.58%	3.636 ± 0.78%
200000	93 ± 3.45%	3.541 ± 0.12%
300000	94 ± 3.87%	2.648 ± 0.97%
400000	94 ± 2.34%	2.133 ± 0.34%
500000	95 ± 2.64%	1.658 ± 0.98%
600000	95 ± 2.32%	1.836 ± 0.12%
700000	96 ± 1.63%	1.326 ± 0.41%
800000	96 ± 1.35%	0.867 ± 0.09%
900000	97 ± 0.27%	0.436 ± 0.34%
1000000	98 ± 0.22%	0.124 ± 0.89%

Using network connection data anomaly detection can be very effective against single and multi-connection-based port scan and DDoS attacks. This shows that there are no natural limits on detection capabilities as in [16, 20, 23, 25] when cooperation of collection and detection on the ad-hoc operating layers is achieved. With our multilayered data collection and detection, anomaly detection performance and particularly the detection rate is high in DSR (99 ± 0.12%), AODV(99 ± 0.01%), and DSDV(98 ± 0.22%). Even when the mobility level was changed, the low false alarm rate remained constant in DSR (0.011 ± 0.34%), AODV (0.561 ± 0.12%), DSDV (0.124 ± 0.89%).

It is obvious from the above results that the anomaly detection performs well in all the three routing protocols. It was expected that DSR and AODV would behave slightly better since anomaly detection works better on a routing protocol in which a degree of redundancy exists within its infrastructure [5, 7, 22]. For example, the DSR and AODV route updates depend on the traffic demand, which makes it possible to establish relationships between the routing activities and the traffic pattern. In contrast, DSDV has a very weak correlation between control traffic and data traffic, even when the traffic feature is preserved. Coming to a conclusion, our approach seems to perform well on both on-demand and proactive protocols. This is due to a carefully selected secret function $f(k)$ (Eq. 7).

Several $f(k)$ functions were selected for our experiments; functions approaching the y-axis of the xy-plane; functions approaching the x-axis of the xy-plane; and functions dividing the xy-plane approximately into two halves. It was noticed that functions approaching the x-axis performed slightly better for all three protocols since route caches, traffic patterns and movements generated data models with small y-values.

5 Conclusions

Since ad-hoc networks can be formed, merged together or partitioned into separate networks on the fly, it is essential to detect intrusions by malicious and compromised nodes during the network's normal operation. In an ad-hoc network, it is also critical to inform neighbouring nodes for their potential compromised status. Due to the different nature of such networks, an intrusion detection component designed to operate in an ad-hoc node should not introduce new weaknesses to the system, should run continuously and should remain transparent to the system.

Our proposed intrusion detection framework is enabled with the main operations of ad-hoc networking at the OSI link and network layers. It makes use of local agents that collect, analyze audit data and distribute a compromised status to the neighbouring nodes for further assessment. The *collection* and *detection* modules use tree structures and Lagrange polynomial interpolation to assemble and discover intrusions from malicious and/or compromised nodes in the ad-hoc network. The *alert* module utilizes linear threshold schemes to inform neighbouring nodes for their status. Our only assumption in the alert module is that the computation of discrete logs in a finite field is intractable.

The main requirements of anomaly detection models and intrusion detection systems in general is a low false positive rate, calculated as the percentage of normalcy variations detected as anomalies, and a high true positive rate, calculated as the percentage of anomalies detected. It was shown that our approach has a high true positive rate (min = 98%) and a very low false positive rate (max = 0.5%) in both proactive and on-demand routing protocols, such as DSR, AODV and DSDV.

We believe that LIDF performs well since we take advantage of the main operations of ad-hoc networking in a layered approach and it will have a positive impact in the field of intrusion detection for ad-hoc networks. In the future, we intent to integrate our approach to hybrid ad hoc and heterogeneous networks in combination with biometric templates for intrusion detection.

References

[1] Bhargava S., Agrawal D., "Security Enhancements in AODV Protocol for Wireless Ad-hoc Networks," Proc. IEEE Vehicular Tech. Conf., IEEE CS Press, 2001, Page(s): 2143–2147.

[2] Bo Sun; Kui Wu; Pooch, U.W., "Towards adaptive intrusion detection in mobile ad-hoc networks", *IEEE Global Telecommunications Conference* (GLOBECOM '04), Volume 6, 29 Nov.-3 Dec. 2004, Page(s):3551 – 3555, (DOI: 10.1109/GLOCOM.2004.1379027)

[3] Brutch, P., Ko, C., "Challenges in intrusion detection for wireless ad-hoc networks", *Symposium on Applications and the Internet Workshops*, 27-31 Jan. 2003, Page(s):368 – 373.

[4] Buchegger S., Le Boudec J. Y., "Performance Analysis of the CONFIDANT Protocol (Cooperation of Nodes: Fairness in Dynamic Ad-Hoc Networks)," *Proceedings of the 3rd International Symposium in Mobile Ad-hoc Networking and Computing*, ACM Press, 2002, pp. 226–236.

[5] Chen, T.M., Venkataramanan, V., "Dempster-Shafer theory for intrusion detection in ad-hoc networks", *IEEE Internet Computing*, Volume 9, Issue 6, Nov.-Dec. 2005, Page(s):35 – 41, (DOI: 10.1109/MIC.2005.123)

[6] Douligeris C., Mitrokosta A., "DDoS attacks and defense mechanisms: classification and state-of-the-art", *Computer Networks: The International Journal of Computer and Telecommunications Networking*, Vol: 44, Issue 5, 2004, Page(s): 643 – 666.

[7] Hasswa, A., Zulkernine, M., Hassanein, H., "Routeguard: an intrusion detection and response system for mobile ad-hoc networks", *IEEE International Conference on Wireless And Mobile Computing, Networking And Communications* (WiMob'2005), Volume 3, 22-24 Aug. 2005, Page(s):336 – 343, (DOI:10.1109/WIMOB.2005.1512922)

[8] Hijazi, A., Nasser, N., "Using mobile agents for intrusion detection in wireless ad-hoc networks" *Second IFIP International Conference on Wireless and Optical Communications Networks* (WOCN 2005), 6-8 March 2005,

Page(s):362 – 366, (DOI: 10.1109/WOCN.2005.1436049)

[9] Jianhua Song, Fan Hong, Yajun Guo, "A Distributed Monitoring Mechanism for Mobile Ad-hoc Networks", *8th International Symposium on Parallel Architectures,Algorithms and Networks* (ISPAN 2005), 07-09 Dec. 2005, Page(s):236 – 240, (DOI:10.1109/ISPAN.2005.6)

[10] Kargl F., Klenk A., Weber M., Schlott S., "Sensors for Detection of Misbehaving Nodes in MANETs," *Proceedings of Detection of Intrusion and Malware and Vulnerability Assessment*, 6-7 July, Germany 2004.

[11] Komninos N., Vergados D., Douligeris C., "Layered Security Design for Mobile Ad-Hoc Networks", *Journal in Computers & Security*, Elsevier Press, Volume 25, Issue 2, 2005, Page(s) 124-134.

[12] Komninos N., Vergados D., Douligeris C., "Detecting Unauthorized and Compromised Nodes in Mobile Ad-Hoc Networks", *Journal in Ad Hoc Networks*, Elsevier Press, Volume 5, Issue 3, 2006, Page(s) 289-298.

[13] Marti S., Giuli T.J., Lai K., and Baker M., "Mitigating Routing Misbehavior in Mobile Ad-hoc Networks" *Proceedings of the 6th annual International Conference in Mobile Computing and Networking*, ACM Press, 2000, Page(s) 255–265.

[14] Menezes J. A., Vanstone A. S., and Van Oorschot C. P., *Handbook of Applied Cryptography*, CRC Press, Inc., USA, 2001.

[15] Mitrokotsa A., Komninos N., Douligeris C., "Intrusion Detection & Response in Ad-hoc Networks", *Special Issue on Advances on Ad Hoc Network Security*, N. Komninos (editor), *International Journal on Computer Research, Nova Science Publishing Inc.*, (to appear), 2007.

[16] Mishra, A., Nadkarni, K., Patcha, A., "Intrusion detection in wireless ad-hoc networks", *IEEE Wireless Communications*, Volume 11, Issue 1, Feb 2004 Page(s):48 – 60, (DOI: 10.1109/MWC.2004.1269717)

[17] Nadkarni, K., Mishra, A., **"**A novel intrusion detection approach for wireless ad-hoc networks", *IEEE Wireless Communications and Networking Conference* (WCNC. 2004), Volume 2, 21-25 March 2004, Page(s):831 – 836.

[18] Patcha, A., Park, J.-M, "A game theoretic approach to modeling intrusion detection in mobile ad-hoc networks", *Proceedings from the Fifth Annual IEEE SMC Information Assurance Workshop*, 10-11 June 2004, Page(s):280 – 284, (DOI:10.1109/IAW.2004.1437828)

[19] Perkins C. E., Belding-Royer E. M., and Chakeres I., "Ad Hoc On-Demand Distance-Vector Routing (AODV)", *IETF Internet Draft*, Oct. 2003.

[20] Stamouli, L., Argyroudis, P.G., Tewari, H., "Real-time intrusion detection for ad-hoc networks", *Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks* (WoWMoM 2005), 13-16 June 2005, Page(s):374 - 380 (DOI:10.1109/WOWMOM.2005.85)

[21] Subhadrabandhu, D., Sarkar, S., Anjum, F., "A framework for misuse detection in ad-hoc Networks-part I", *IEEE Journal on Selected Areas in Communications*, Volume 24, Issue 2, Feb. 2006 Page(s):274 – 289, (DOI:10.1109/JSAC.2005.861387)

[22] Sun, B., Wu, K., Pooch, U.W., "Routing anomaly detection in mobile ad-hoc networks", *The 12th International Conference on Computer Communications and Networks* (ICCCN 2003), 20-22 Oct. 2003, Page(s):25 – 31, (DOI: 10.1109/ICCCN.2003.1284145)

[23] Thamilarasu, G., Balasubramanian, A., Mishra, S., Sridhar, R., "A cross-layer based intrusion detection approach for wireless ad-hoc networks", *IEEE*

International Conference on Mobile Adhoc and Sensor Systems, Nov. 7, 2005, Page(s):855 – 861, (DOI:10.1109/MAHSS.2005.1542882)

[24] Watkins, D., Scott, C., "Methodology for evaluating the effectiveness of intrusion detection in tactical mobile ad-hoc networks", *IEEE Wireless Communications and Networking Conference* (WCNC. 2004), Volume 1, 21-25 March 2004, Page(s):622 – 627.

[25] Yan Xia, Ren-Fa Li, Ken-Li Li, "Intrusion detection using mobile agent in ad-hoc networks", *Proceedings of 2004 International Conference on Machine Learning and Cybernetics*, Volume 6, 26-29 Aug. 2004, Page(s):3383 – 3388.

[26] Yu Liu, Yang Li, Hong Man, "MAC layer anomaly detection in ad-hoc networks" *Proceedings from the Sixth Annual IEEE Systems, Man and Cybernetics (SMC) Information Assurance Workshop*, 15-17 June 2005, Page(s):402 – 409, (DOI:10.1109/IAW.2005.1495980)

[27] Zhang Y., Lee W., "Intrusion detection in wireless ad-hoc networks", *Proceedings of the 6th annual international conference on Mobile computing and networking*, Page(s) 275-283, Boston, Massachusetts, United States, 2000.

[28] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwary, H. Yang, S. Zhou, "Specification-based Anomaly Detection: A new approach for Detecting Network Intrusions", *Proceedings of the 9th ACM Conference on Computer and Communication Security*, Page(s): 265-274, 2002.

[29] James C. Reynolds, James Just, Larry Clough, Ryan Maglich, "On-Line Intrusion Detection and Attack Prevention Using Diversity, Generate-and-Test, and Generalization", *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03)*, 2003